CPSC 1155 – Assignment 3

Math, Characters, Strings, and Files

Objectives

The goal of this assignment is to practice with Characters and Strings in C++ programs.

Readings

You should be reading chapter 4 of the textbook. The lectures and labs will provide additional supporting material.

Instructions

For each **Problem Statement**, follow the steps below:

1. Read the problem statement and clarify the problem.
   a. Break the problem into smaller problems if needed.
2. Determine the IPO.
   a. Determine input, output, intermediate variables, constants, and conditions.
   b. Declare the variables and constants (data type + meaningful names).
   c. Work out the problem by hand using typical input values. Determine the range of valid input values.
   d. Determine the process.
3. Write a pseudocode as required.
4. Write a C++ program (use the given filename) that implements the pseudocode.
   a. Add comments where needed. Make sure to use a comments header to reflect the intention of your program and name of the author (you) and the date the program was written.
   b. Test, debug, and execute the program using typical values.

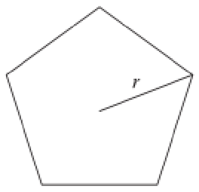
Submit according to the instruction in the "Submission" section.

Problem Statements

1. (pentagon_area.cpp) Write a **C++ program** that prompts the user to enter the length from the center of a pentagon to a vertex and computes the area of the pentagon, as shown in the following figure.

   The formula for computing the area of a pentagon is $Area = \frac{5 * s^2}{4 * \tan(\frac{\pi}{5})}$, where s is the length of a side. The side can be computed using the formula $s = 2r\ \sin\frac{\pi}{5}$, where r is the length from the center of a pentagon to a vertex. Round up two digits after the decimal point. Display error message if input is invalid. Here is a sample run:

```
Enter the length from the center to a vertex: 5.5
The area of the pentagon is 71.9236
```



2. (order_fruits.cpp) Write an **algorithm** and **C++ program** that prompts the user to enter three fruits and displays them in descending order.

   Note: An algorithm basically describes your solutions in steps, so it does not have to follow a specific syntax.

   Hint: Transform the input names to capitalize the first letter of each string, and then compare them. If the words have more than 1 word, only the first letter of the first word needs to be capitalized.

   Here is a sample run:

   ```
   Enter the first fruit: peach
   Enter the second fruit: pear
   Enter the third fruit: Star Fruit
   The three fruits in alphabetical descending order are Star Fruit, pear, peach
   ```

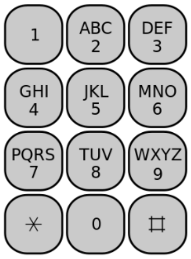
3. (decimal_to_hex.cpp) Write a **pseudocode** and **C++ program** that prompts the user to enter an integer between 0 and 15 and displays its corresponding hex number. Here are some sample runs:

   ```
   Enter a decimal value (0 to 15): 11
   The hex value is B
   ```

   ```
   Enter a decimal value (0 to 15): 5
   The hex value is 5
   ```

   ```
   Enter a decimal value (0 to 15): 31
   31 is an invalid input
   ```

4. (key_pads.cpp) The international standard letter/number mapping found on the telephone is shown below. Write a **C++ program** that prompts the user to enter a letter (must be converted to uppercase) and displays its corresponding number. [Hint: You may use a switch statement.]

   Here are some sample runs:

   ```
   Enter a letter: A
   The corresponding number is: 2
   ```

   ```
   Enter a letter: g
   The corresponding number is: 4
   ```

   ```
   Enter a letter: +
   + is an invalid input
   ```

5. (major_and_year.cpp) Write a **C++ program** that prompts the user to enter two characters and displays the major and status represented in the characters. The first character indicates the major and the second is number character 1, 2, 3, 4, which indicates whether a student is a freshman, sophomore, junior, or senior. Suppose the following characters are used to denote the majors:

   M: Mathematics
   C: Computer Science
   I: Information Technology

   Here are sample runs:

   ```
   Enter two characters: M1
   Mathematics Freshman
   ```

```
Enter two characters: C3
Computer Science Junior

Enter two characters: T3
Invalid major code

Enter two characters: M7
Invalid status code
```

6. (check_ssn.cpp) Write a **C++ program** that prompts the user to enter a Social Security Number in the format ddd-dd-dddd, where d is a digit, and checks the validity of the SSN (The input may contain either digit or dash and conforms to the mentioned format).

   Here are sample runs:

```
Enter SSN: 232-23-4532
232-23-4532 is valid

Enter SSN: 32-123-4532
32-123-4532 is invalid
```

7. (format_output.cpp) Write a **C++ program** that prompts the user to enter three names. The program generates an ID for each user with three random uppercase characters and 3 random digits in the form of CCC-DDD (C for character, and D for digit). [Hint: The digits could be declared as characters.]

   The program displays the information in the following format:

Name	Marie Curie	Thomas Edison	Albert Einstein
ID	MCF-198	THM-397	EZA-599

8. (format_output_text.cpp) Use the code for question 7 and write a **C++ program** that creates a text file with an output similar to question 7. You can use string manipulators such as setw and setprecision for an output file.

   Write another **C++ program** to open the text file and displays the names and IDs in the file with the same format as question 7.

Submission

Submit a zip folder named as yourName_Assign3.zip to Brightspace. This folder should consist of the **C++ codes** in individual .cpp files and one pseudocode.txt file with all your **pseudocode** and **algorithm** (for questions 2 and 3). Please name each cpp file exactly as the instructed names at the beginning of each question.

Please make sure that all your .cpp files compile and run properly before submission. Your file must run properly in order to receive full marks.

Assignments are individual work.

Marking Scheme

There are 10 marks for each question with the following details:

Question 1: pentagon_area.cpp

- 2 for checking boundary

- 8 for correct solution

Question 2: order_fruits.cpp

- 2 for capitalized first letter before comparison
- 4 for algorithm
- 4 for correct solution

Question 3: decimal_to_hex.cpp

- 2 for checking boundary
- 4 for pseudocode
- 4 for correct solution

Question 4: key_pads.cpp

- 2 for checking boundary
- 8 for correct solution

Question 5: year_and_major.cpp

- 2 for checking major code
- 2 for checking status code
- 6 for correct solution

Question 6: check_ssn.cpp

- 10 for correct solution

Question 7: format_output.cpp

- 5 for correct display
- 5 for correct random generation

Question 8: format_output_text.cpp

- 10 for correct solution